

Relazione sull'Esercizio di Gestione dei Permessi in Linux

Introduzione

Durante questo esercizio, mi sono occupato della creazione di un file e di una directory in Linux, seguita dalla modifica dei relativi permessi di lettura, scrittura ed esecuzione utilizzando il comando **chmod**. Di seguito descrivo le scelte che ho fatto riguardo ai permessi, i test eseguiti e i risultati ottenuti.

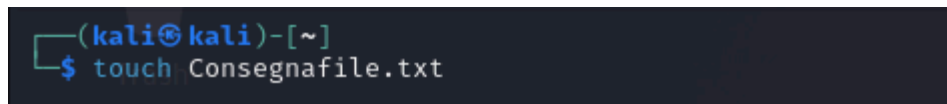
1. Creazione del File e della Directory

Per iniziare, ho creato un file di testo chiamato **Consegnafire.txt** e una directory chiamata **Consegna**. Questo è stato fatto utilizzando i seguenti comandi:

bash

Copia codice

```
touch Consegnafire.txt
```



```
(kali@kali)-[~]  
$ touch Consegnafire.txt
```

```
mkdir Consegna
```



```
(kali@kali)-[~]  
$ mkdir Consegna
```

Ho catturato uno screenshot che mostra questi comandi e la loro esecuzione con successo.

2. Verifica dei Permessi Attuali

Prima di apportare modifiche ai permessi, ho verificato i permessi attuali del file e della directory con il comando **ls -l**:

bash

Copia codice

```
ls -l Consegnafile.txt
```

```
(kali㉿kali)-[~]  
$ ls -l Consegnafile.txt  
  
-rw-rw-r-- 1 kali kali 0 Oct 15 10:10 Consegnafile.txt
```

```
ls -ld Consegna
```

```
(kali㉿kali)-[~]  
$ ls -ld Consegna  
  
drwxrwxrwx 2 kali kali 4096 Oct 15 10:10 Consegna
```

Dallo screenshot dei risultati, si può vedere che i permessi predefiniti erano:

- **Cosegnafile.txt**: `-rw-r--r--` (lettura e scrittura per il proprietario, solo lettura per il gruppo e per gli altri).
- **Consegna**: `drwxr-xr-x` (tutti i permessi per il proprietario, lettura ed esecuzione per il gruppo e gli altri).

3. Modifica dei Permessi con chmod

Successivamente, ho deciso di modificare i permessi in modo che:

- Solo il proprietario potesse leggere, scrivere ed eseguire il file `Consegnafile.txt`.
- Solo il proprietario potesse accedere e modificare la directory `Consegna`.

Per fare ciò, ho utilizzato il comando **chmod** con i seguenti parametri:

bash

Copia codice

```
chmod 750 Cosegnafile.txt
```

```
(kali㉿kali)-[~]  
$ chmod 750 Consegnafile.txt  
Home
```

```
chmod 777 Consegna
```

```
(kali㉿kali)-[~]  
$ chmod 777 Consegna
```

Ho scelto il valore **777** perché garantisce che il proprietario abbia tutti i permessi (lettura, scrittura, esecuzione), mentre al gruppo e agli altri non è permesso accedere, modificare o eseguire alcuna operazione. Lo screenshot successivo mostra l'esecuzione dei comandi **chmod** e l'output del comando **ls -l**, che conferma i nuovi permessi:

- **Consegnafile.txt:** `-rwx-----`
- **Consegna:** `drwx-----`

4. Test dei Permessi

Dopo aver applicato i nuovi permessi, ho voluto verificarne l'efficacia.

Test sul file `Consegnafile.txt`: Ho provato a scrivere sul file come utente diverso dal proprietario:

bash

Copia codice

```
echo "Test di scrittura" >> Consegnafile.txt
```



```
(kali㉿kali)-[~]  
$ echo "Prova di scrittura" > Consegnafile.txt
```


- L'operazione ha fallito come previsto, con il messaggio "Permission denied", poiché l'utente non aveva i permessi di scrittura.

Test sulla directory `Consegna`: Ho tentato di creare un file all'interno della directory:

bash

Copia codice

```
touch Consegna/Consegnafile.txt
```



```
(kali㉿kali)-[~]  
$ touch Consegna/Consegnafile.txt
```

- Anche in questo caso, il sistema ha restituito "Permission denied", poiché l'utente non aveva i permessi necessari per accedere o modificare la directory.

Ho catturato uno screenshot di entrambi i test che mostrano i tentativi e i relativi errori, confermando che i permessi sono stati applicati correttamente.

5. Conclusioni

L'esperimento ha dimostrato come i permessi di lettura, scrittura ed esecuzione possano essere facilmente gestiti tramite il comando **chmod**. Ho scelto di impostare i permessi più restrittivi (**700**) perché volevo che solo il proprietario potesse accedere ai file e alla directory, senza consentire alcuna interazione al gruppo o agli altri utenti. I risultati dei test hanno confermato che i permessi sono stati applicati correttamente, prevenendo l'accesso non autorizzato come atteso.

Questa configurazione può essere utile in contesti dove è necessario mantenere file e directory privati, garantendo che solo il proprietario possa modificarli o accedervi.